

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平11-7451

(43) 公開日 平成11年(1999) 1月12日

(51) Int.Cl.<sup>6</sup>

G 0 6 F 17/30

識別記号

F I

G 0 6 F 15/419

15/40

3 1 0

3 7 0 J

審査請求 未請求 請求項の数 9 O L (全 22 頁)

(21) 出願番号

特願平9-161458

(22) 出願日

平成9年(1997) 6月18日

(71) 出願人 000005496

富士ゼロックス株式会社

東京都港区赤坂二丁目17番22号

(72) 発明者 梅基 宏

神奈川県足柄上郡中井町境430 グリーン

テクなかい 富士ゼロックス株式会社内

(72) 発明者 館野 昌一

神奈川県足柄上郡中井町境430 グリーン

テクなかい 富士ゼロックス株式会社内

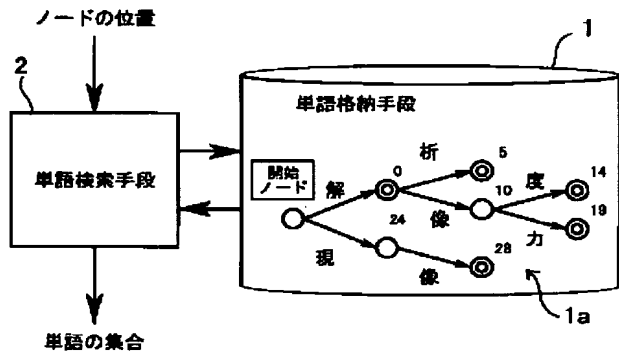
(74) 代理人 弁理士 服部 毅巖

(54) 【発明の名称】 単語検索装置及び単語検索プログラムを記録した媒体

(57) 【要約】

【課題】 少ない記憶容量で高速に単語の検索が行えるようにする。

【解決手段】 単語格納手段1には、単語の集合1aが、深さ優先順にノードが記録されるトライ形式（トライ・インデックス）で格納されている。単語検索手段2は、単語格納手段1における格納先の位置情報が入力されると、単語格納手段1のトライを根から順にたどっていき、入力された位置のノードをまでの経路を求め、求められた経路以降の全ての経路をたどって到達するノードに対応する全ての単語を取得し、取得した単語集合を出力する。これにより、ノード位置から、そのノードを含む単語の集合を得ることができる。しかも、トライ中のノードを指定されたときに、親のノードへのリンク情報を用いることなく、そのノードを含む経路を特定することができる。



1

## 【特許請求の範囲】

【請求項 1】 単語集合から単語を検索する単語検索装置において、

深さ優先順にノードが記録されるトライ形式にしたがって、ノードに対応付けられた単語の集合が格納された単語格納手段と、

前記単語格納手段におけるノードの位置が入力されると、前記単語格納手段のトライを根から順にたどっていき、入力された位置のノードまでの経路を求め、求められた経路以降の全ての経路をたどって到達するノードに対応する全ての単語を取得し、取得した単語の集合を出力する単語検索手段と、

を有することを特徴とする単語検索装置。

【請求項 2】 前記単語格納手段に含まれる単語に対応するキーと、各単語を構成しているノードの位置とを対応付けて格納するキーインデックス格納手段と、前記キーインデックス格納手段中の任意のキーが入力されると、前記キーインデックス格納手段から、入力されたキーに対応するノードの位置の集合を取得し、取得したノードの位置の集合を前記単語検索手段に対して出力するノード位置検索手段と、

をさらに有することを特徴とする請求項 1 記載の単語検索装置。

【請求項 3】 前記キーインデックス格納手段は、前記単語格納手段に含まれる単語を構成する全ての文字と、各文字を表しているノードの位置とを対応付けていることを特徴とする請求項 2 記載の単語検索装置。

【請求項 4】 前記キーインデックス格納手段は、前記単語格納手段に含まれる単語を構成する文字のうち、単語の先頭文字および末尾文字を除いたすべての文字と、各文字を表しているノードの位置とを対応付けていることを特徴とする請求項 3 記載の単語検索装置。

【請求項 5】 正規表現が入力されると、入力された正規表現を解析し、前記ノード位置検索手段に対し、正規表現中の文字を渡してノードの位置の集合を受け取ると共に、前記単語検索手段に対し、ノードの位置の集合を渡して単語の集合を受け取り、入力された正規表現に合致する単語の集合を出力する正規表現解析手段を更に有することを特徴とする請求項 3 記載の単語検索装置。

【請求項 6】 前記単語格納手段は、単語と各単語に対応する情報へのポインタとを組にして格納しており、前記単語検索手段は、前記単語格納手段から各々のノードが表す単語および単語に対応する情報へのポインタの集合を出力することを特徴とする請求項 1 記載の単語検索装置。

【請求項 7】 前記単語格納手段は、深さ優先順にノードが記録されるトライ形式にしたがって、ノードに対応付けられた単語の集合を格納していると共に、単語と各単語に関連する関連単語のノードの位置の集合とを対応付けて格納しており、

2

単語が入力されると、前記単語格納手段から、入力された単語に関連する単語のノードの位置の集合を取得し、取得したノードの位置の集合を前記単語検索手段に対して出力するノード位置検索手段を更に有することを特徴とする請求項 1 記載の単語検索装置。

【請求項 8】 前記単語格納手段は、表記によって表された表記単語の集合を、深さ優先順にノードが記録されるトライ形式で格納していると共に、表記単語と各表記単語に対応するの文字列を構成しているよみ単語のノードの位置の集合とを対応付けて格納するよみインデックス格納手段と、よみによって表された単語の集合を、深さ優先順にノードが記録されるトライ形式で格納していると共に、よみ単語と各よみ単語に対応する表記単語の文字列を構成しているノードの位置の集合とを対応付けて格納している表記インデックス格納手段と、から構成されていることを特徴とする請求項 7 記載の単語検索装置。

【請求項 9】 コンピュータに単語集合から単語を検索させるための単語検索プログラムを記録した媒体において、

深さ優先順にノードが記録されるトライ形式にしたがって、ノードに対応付けられた単語の集合が格納された単語格納手段、

前記単語格納手段におけるノードの位置が入力されると、前記単語格納手段のトライを根から順にたどっていき、入力された位置のノードまでの経路を求め、求められた経路以降の全ての経路をたどって到達するノードに対応する全ての単語を取得し、取得した単語の集合を出力する単語検索手段、

としてコンピュータを機能させるための単語検索プログラムを記録した媒体。

## 【発明の詳細な説明】

## 【0001】

【発明の属する技術分野】本発明は単語の集合の中から単語につけられたキーをもとに効率的に単語を検索する単語検索装置及び単語検索プログラムを記録した媒体に関し、特に任意の位置にある文字を指定して、関連する単語を高速に検索する単語検索装置及びそのような検索をコンピュータに行わせるための単語検索プログラムを記録した媒体に関する。

## 【0002】

【従来の技術】何らかのキーから単語を検索するという処理は、辞書の検索やかな漢字変換といったテキスト情報処理システムにおける基本的な過程である。それだけに、キーから単語を検索するために必要な処理速度や記憶容量は、そのような処理システム全体の性能を大きく左右する。したがって、このような処理をより高速に、かつより少ない記憶容量から実現することによって、非常に大きな実用的効果を得ることができる。

【0003】さて、単語をキーとして文書を検索するシ

## 3

システムはすでに多く存在しているが、このような文書検索システムにおいても、検索の結果として単語を出力する機能は、必須ではないが非常に有効な場合がある。

【0004】文書検索システムにおいて多くの場合、キーである単語は、文書中に含まれているか、もしくはシステムの管理者や文書の作成者によってあらかじめ選定される。この場合、検索システムを利用する立場のユーザからは、登録してあるキーの単語が何であるか分からないことがしばしばある。そのため、ユーザを支援するために、文書ではなくキーワードを何らかの方法で検索

【0005】キーワードを検索する具体的な方法として、たとえば、よみから表記の単語を検索する方法、または、ある文字もしくは文字列を含む単語を検索する方法、さらには、任意の正規表現を満たす単語を検索する方法などが考えられる。

【0006】また、キーワードによる文書検索システムにおいては、単語から所望の文書もしくは文書へのポインタを高速に得るために、インデックスとしてトライと呼ばれる木構造（トライ・インデックス）を用いることが多い。このトライを用いれば、高速に単語の検索を行うことができる。トライから単語を検索するときには、ほぼ入力文字列の長さ に 比例する程度の処理ステップ数しか必要としない。またデータ圧縮率も比較的良好いため、トライは大量の索引単語を格納するという用途に向いている。加えて、トライを用いる場合、単語の先頭部分の文字列を指定すると、その文字列から始まるすべての単語を、簡単な処理によって求めることができるという利点もある。

【0007】ところが、トライから、単語の先頭以外の任意の位置の部分文字を含む単語を、高速に探し出すことはできない。ここで、これらの場合には、単語を表すトライとは別個にインデックスを設けることによって、任意の位置の部分文字から、その文字を含む単語を検索することが行われている。

【0008】いま、文字を検索キーとして、その文字から綴りの中の任意の位置を含む単語を検索する場合を考えることにしよう。単語は複数の文字によって構成されているので、ある1つの単語は、複数の検索キー（文字）に対応付けられている。この場合のように、検索対象が複数のキーにリンクされているときは、検索対象の集合を1つのデータ構造の中に格納し、検索キーにはそのデータ構造中のポインタの値を対応させると、必要な記憶容量は少なく済む。

【0009】そこで、単語集合のデータを少ない記憶容量で表すことができ、かつ、ポインタによって単語を特定できるデータ構造について以下に考察する。単語の集合を格納するデータ構造として一般的なのは、固定長または可変長の文字列として格納するレコード構造である。このレコード構造を用いた単語検索方式を「第1の

## 4

従来例」と呼ぶことにする。このレコード構造であれば、任意の単語に対して、単語の総数によらずほぼ一定の時間で高速にアクセスすることができる。したがって、単語集合をこのようなレコード構造として格納することによって、単語検索システムにおける高速なキーワード検索を実現することができる。

【0010】また、第1の従来例とは別に、トライ・インデックスを単語集合のデータとみなして、単語の末尾に対応するトライ中のノードの識別番号を単語へのポインタとすることが考えられる。これを「第2の従来例」と呼ぶことにする。トライのような木構造においては、根を除くすべてのノードの親ノードは一意に存在する。したがって、親ノードへのリンク情報をすべてのノードに持たせることによって、1つのノードを指定したときに、そのノードから根に至るまでの経路は一意に決定することができる。

【0011】

【発明が解決しようとする課題】しかし、単語をキーとするトライ・インデックスが存在し、なおかつそのインデックス中の単語を何らかのキーから検索する場合において、上記の従来の方式には以下のような問題点があった。

【0012】上記の第1の従来例、すなわち、単語の集合を固定長または可変長の文字列として格納するレコード構造のデータを、インデックスとは別個に用意する方法では、キーから単語への高速な検索は実現できるが、すでにトライ形式で格納されている単語の集合のデータとは別個にあらたなデータが必要となる。そのために必要な記憶容量は無視し得ないほど大きくなってしまいう問題点がある。

【0013】上記の第2の従来例、すなわち、トライ・インデックスを単語集合のデータとみなして、単語の末尾に対応するトライ中のノードの識別番号を単語へのポインタとする方法では、第1の従来例と比べて単語集合を表すためのデータが不要な分だけ、キーから単語への検索自体に必要な記憶容量は第1の従来例に比べて少なくて済むが、トライ・インデックスに対して本来ならば不要な、親ノードへのリンク情報を追加することになってしまう。したがって、検索システム全体としてみた場合、第2の従来例は第1の従来例に比べて、記憶容量の面で著しく改善されているとは言えない。

【0014】本発明はこのような点に鑑みてなされたものであり、少ない記憶容量で高速に単語を検索できる単語検索装置を提供することを目的とする。また、本発明の他の目的は、コンピュータに対して、少ない記憶容量で高速に単語を検索させるための単語検索プログラムを記録した媒体を提供することである。

【0015】

【課題を解決するための手段】本発明では上記課題を解決するために、単語集合から単語を検索する単語検索装

5

置において、深さ優先順にノードが記録されるトライ形式にしたがって、ノードに対応付けられた単語の集合が格納された単語格納手段と、前記単語格納手段におけるノードの位置が入力されると、前記単語格納手段のトライを根から順にたどっていき、入力された位置のノードをまでの経路を求め、求められた経路以降の全ての経路をたどって到達するノードに対応する全ての単語を取得し、取得した単語の集合を出力する単語検索手段と、を有することを特徴とする単語検索装置が提供される。

【0016】この単語検索装置によれば、単語格納手段における格納先の位置情報が入力されると、単語検索手段が、単語格納手段のトライを根から順にたどっていき、入力された位置のノードまでの経路を求め、求められた経路以降の全ての経路をたどって到達するノードに対応する全ての単語を取得し、取得した単語の集合が出力される。その結果、入力された位置に対応する単語を含む複数の単語を、高速に検索することができるとともに、単語格納手段に必要な記憶容量は少なくすむ。

【0017】また、コンピュータに単語集合から単語を検索させるための単語検索プログラムを記録した媒体において、深さ優先順にノードが記録されるトライ形式にしたがって、ノードに対応付けられた単語の集合が格納された単語格納手段、前記単語格納手段における格納先の位置情報が入力されると、前記単語格納手段のトライを根から順にたどっていき、入力された位置のノードをまでの経路を求め、求められた経路以降の全ての経路をたどって到達するノードに対応する全ての単語を取得し、取得した単語の集合を出力する単語検索手段、としてコンピュータを機能させるための単語検索プログラムを記録した媒体が提供される。

【0018】この媒体に記録された単語検索プログラムをコンピュータに実行させることにより、深さ優先順にノードが記録されるトライ形式にしたがって、ノードに対応付けられた単語の集合が格納された単語格納手段としての機能と、単語格納手段における格納先の位置情報が入力されると、単語格納手段のトライを根から順にたどっていき、入力された位置のノードをまでの経路を求め、求められた経路以降の全ての経路をたどって到達するノードに対応する全ての単語を取得し、取得した単語の集合を出力する単語検索手段としての機能とが、コンピュータによって実現される。

【0019】

【発明の実施の形態】以下、本発明の実施の形態を図面を参照して説明する。図1は、本発明の原理構成図である。本発明に係る単語検索装置は、単語格納手段1と単語検索手段2とを有している。

【0020】単語格納手段1には、単語の集合1aが、深さ優先順にノードが記録されるトライ形式（トライ・インデックス）で格納されている。ここで、ノードの深さとは、トライの根からそのノードまでの経路の長さで

6

ある。そして、ノードの深さ優先順とは、できる限り探索を深さ方向に進めたときにたどるノードの順番である。あるノードの子孫のノードと、弟のノードとを比較した場合、常に、子孫のノードの方が優先順が高い（アドレスの値が小さい）。また、兄弟同士で比べると、兄となるノードの方が優先順が高い。

【0021】単語検索手段2は、単語格納手段1における格納先の位置情報が入力されると、単語格納手段1のトライを根から順にたどっていき、入力された位置のノードまでの経路を求め、求められた経路以降の全ての経路をたどって到達するノードに対応する全ての単語を取得し、取得した単語集合を出力する。ノードの位置からそのノードを含む単語または単語の集合を求めるアルゴリズムを以下に記述する。

【0022】図2は、ノードの位置からそのノードを含む単語または単語の集合を求めるアルゴリズムを示すフローチャートである。これは、ノードの位置情報を受け取った単語検索手段2が行う処理である。

【S1】単語格納手段1のトライにおいて、開始ノードから遷移することを考える。まず、トライにおけるラベルの列を格納するために空のラベルスタックを用意し、開始ノードを「ノードA」とおき、与えられたノード位置に相当するノードを「ノードX」とおく。

【S2】ノードAの長子のノードを「B1」とする。ノードAの次の位置に存在するため、長子のノードであるノードB1は直ちに求まる。

【S3】ノードB1の隣接する弟ノードをノードB2とする。トライを根からたどることによって、任意のノードの隣接する弟のノードB2も直ちに求まる。

【S4】ノードXがノードB1と等しいか否かを判断する。等しければステップS8に進み、等しくなければステップS5に進む。

【S5】ノードXがノードB2よりも前か否かを判断する。前であるならステップS6に進み、前でないならステップS6に進む。

【S6】ノードXがノードB2よりも前にある場合、ノードXは、ノードB1までの経路をたどることが分かる。そこで、ノードAからノードB1に至るまでのアークに付加されたラベルをラベルスタックに格納し、ノードAにノードB1を代入して、ステップS2に戻る。

【S7】ノードXがノードB2以降にある場合、ノードXが、ノードB1の経路をたどらないことが分かる。そこで、ノードB1にノードB2を代入して、ステップS3に戻る。

【S8】ノードXがノードB1と等しい場合、ノードB1以降の全ての経路をたどり、それらの経路となるアークに付加されたラベル列を求める。

【S9】ラベルスタックに格納されているラベル列とノードB1以降の経路から得られたラベル列の連結し、処理を終了する。これにより、求めるべき単語が表され

10

20

30

40

50

る。

【0023】このような処理を単語検索手段2が行うことにより、ノード位置から、そのノードを含む単語の集合を得ることができる。しかも、トライ中のノードが深さ優先順に記録され、かつトライを根からたどることによって任意のアークに関してその隣接する弟アークが特定できるため、トライ中のノードを指定されたときに、親のノードへのリンク情報を用いることなく、そのノードを含む経路を特定することができる。その理由を以下に説明する。

【0024】まず、前提として、トライ中のあるノードをノードAとし、ノードAの長子のノードをB1とし、ノードB1の隣接する弟ノードをB2とし、ノードB1の長子のノードをCとする。いま、ノードXの位置が指定され、ノードAからノードXに至るまで経路を求めたいとする。このとき、ノードXはノードAからたどれることは分かっているものとする。

【0025】本発明に係るトライでは、ノードは深さ優先順に記録されているので、ノードCは、ノードB1とノードB2との間にある。仮に、ノードXがノードCと等しいとき、ノードXは、ノードB1よりも後にあり、かつ、ノードB2よりも前にあることになる。したがって、ノードXはノードB1からたどれることになる。以上のことから、ノードXがノードB1とノードB2との間にある場合、ノードXにはノードAからノードB1に遷移する経路をたどって到達できることがわかる。

【0026】また、ノードXがノードB2よりも後にある場合、ノードXは、ノードAからノードB1に遷移する経路をたどらないことがわかる。この場合、ノードXにはノードAから、ノードB1の弟のノードのいずれかを遷移する経路をたどって到達できることが分かる。以上の考察を繰り返すことによって、ノードAからノードXまで到達できる経路が分かる。

【0027】以上の結果、ノードの位置を指定されたときに、トライの根から遷移することによって、そのノードを含む単語を求めることができる。しかも、トライにおける単語を、トライにおけるノードの位置をポインタとして参照することによって、トライ・インデックス以外に単語集合を表すデータを設けておく必要がなく、情報の記憶容量が少なくてすむ。

【0028】また、単語に対する検索キー（文字）の入力に応じて、単語の集合が出力されるようにすることもできる。そのような単語検索装置について、以下に説明する。

【0029】図3は、単語へのキーを入力とする単語検索装置の原理構成図である。この単語検索装置は、単語格納手段11、キーインデックス格納手段12、ノード位置検索手段13及び単語検索手段14で構成されている。なお、単語格納手段11と単語検索手段14とは、図1中の単語格納手段1と単語検索手段2と同じ機能有

しているため、ここでは説明を省略する。

【0030】キーインデックス格納手段12は、単語へのキー（文字）と、そのキーと同じラベルが付加されているノードの位置情報とを対応付けて格納している。このように、キーとノード位置との対応関係のすべてをもとにキーインデックス格納手段12が構成されている。なお、1つのキーに対して複数の単語が対応している場合、キーインデックス格納手段12において、そのキーにはノードの位置の集合が対応している。

10 【0031】ノード位置検索手段13は、単語へのキーが入力されると、キーインデックス格納手段12の中から、対応するノードの位置の集合を検索する。そして、得られたノードの位置の集合を単語検索手段14へ入力する。

20 【0032】以後、単語検索手段14が、単語格納手段11から各ノード位置に対応する単語集号を検索し、単語集合を出力する。これにより、単語を構成している文字を検索キーとして入力し、その文字を含む単語の集合を得ることができる。すなわち、単語の先頭以外の任意の位置の部分文字を含む単語の集合を探し出すことができる。

30 【0033】なお、キーインデックス格納手段12は、単語格納手段11に含まれる単語を構成する文字のうち、単語の先頭文字および末尾文字を除いたすべての文字と、単語格納部中でのその文字を表しているノードの位置とを対応付けているものであってもよい。それは、単語の先頭文字あるいは末尾文字から始まる単語は、従来の技術を用いて検索できるため、特にキーインデックス格納手段12において管理する必要がないからである。このように、単語の先頭文字および末尾文字に対応するノードの位置の情報を省略すれば、キーインデックス格納手段12に必要な記憶容量をさらに減らすことができる。

40 【0034】また、単語格納手段11には、単語に対応する情報（例えば、その単語を含む文書）へのポインタとそれに対応する単語とを組にして格納しておき、単語検索手段14は、単語格納手段11から各々のノードが表す単語および単語に対応する情報へのポインタの集合を出力するようにしてよい。これにより、任意のキー（文字）を入力することにより、そのキーを含む単語を取得し、さらに、取得した単語に対応する情報（文書など）を得ることができる。

50 【0035】以上が本発明の基本となる原理構成である。以下に、上記の構成をより具体化した単語検索装置の実施の形態を説明する。図4は、本発明の第1の実施の形態を示すブロック図である。これは、単語を入力として、その単語に関連する単語の集合を検索する単語検索装置である。

【0036】この実施の形態に係る単語検索装置は、関連単語インデックス部21、ノード位置検索部22及び

単語検索部23で構成されている。関連単語インデックス部21には、図3の原理構成で示した単語格納手段11とキーインデックス格納手段12との情報を保持していると同時に、ある単語とその単語に対する関連語集合との対応関係をも保持している。

【0037】ノード位置検索部22は、単語の入力を受けると、その単語に関連する単語（その単語自身も含む）のノードの位置の集合を受け取り、単語検索部23に入力する。単語検索部23は、入力されたノードの位置の集合に基づいて、関連単語インデックス部21を検索し、関連単語の集合を取得し、出力する。

【0038】ここで、本発明の実施の形態の詳細を説明する前に、木に関する用語をあらためて定義する。木は、ノードとよばれる要素の集合に対して階層関係を与えたものである。以下、木において与えられる階層関係は、親子親戚関係を表すことばで表現することにする。木においては、自らを含むすべてのノードを子孫とするノードが1つ存在する。これを開始ノードとよぶことにする。開始ノード以外のすべてのノードに対して、その親であるノードが必ず1つ存在する。ノードとノードとの間の親子関係のつながりを示すものを、アークとよぶことにする。そして、ノードは、特別な状態として、終了状態を持つことができることにし、終了状態ではない他のノードと区別することにする。また、自分自身以外の子孫を持たないノードは、終了状態であるとする。

【0039】トライは木構造の一種であり、開始ノードから終了状態のノードまでを、任意個のノードを経由してアークによって結ばれている経路の1本1本に、集合の中の1つの単語が対応している。本説明文中では、トライにおいて、単語を構成する要素である文字を、アークに対して割りつけることにする。また、1つのノードから派生している各々のアークに対応する文字は、すべて異なるようにノードを構成する。トライにおいては、開始ノードを除いて1つのノードに遷移するアークは必ず1つ存在するので、このようなノードとアークの組を1つの辺節という単位と見なすことにする。以下にある単語の集合を表したトライの例を示す。

【0040】図5は、単語の集合の例を示す図である。この単語集合には、6個の単語「解」「解析」「解像度」「解像力」「現像」「像」がある。それぞれの単語には、ポイントが対応付けられている。ポイントは、その単語を含む文書の識別子集合の位置を指し示すものであるが、この実施の形態では、関連語のノード位置の集合を指し示すのにも用いられる。「解」のポイントは「T1」であり、「解析」のポイントは「T2」であり、「解像度」のポイントは「T3」であり、「解像力」のポイントは「T4」であり、「現像」のポイントは「T5」であり、「像」のポイントは「T6」である。この単語の集合を基に、深さ優先順のノードが記録されたトライを生成する。

【0041】図6は、深さ優先順にノードが記録されたトライの例を示す図である。これは、図5に示した6個の単語「解」「解析」「解像度」「解像力」「現像」

「像」をトライによって表したものである。図中、丸印若しくは2重丸で表しているのがノード30～38である。2重丸は終了状態のノード（対応する単語が存在するノード）を表している。根であるノード30が単語検索時の「開始ノード」となる。終了状態のノードの右下にある記号はその終了状態のノードによって示されることばに関連する単語の集合を意味する識別子をそれぞれ表している。なお、図中の開始ノード以外の各ノード31～38の近傍に表示しているのが、それぞれのノードのアドレス（位置）である。また、各ノード31～38を接続している矢印がアーク41～48である。各アーク41～48の上にある文字（ここでは漢字1文字）はラベルである。

【0042】図6におけるトライのノードを深さ優先順に並べると、根のノード30から派生する「解」のラベルをもつアーク41を遷移してきたノード31、ノード31から派生する「析」のラベルをもつアーク42を遷移してきたノード32、ノード31から派生する「像」のラベルをもつアーク43を遷移してきたノード33、…の順となる。

【0043】このように、辺節を深さ優先順に記録し、かつ、ある辺節からそのすぐとなりの弟の位置が特定できるようにする。図7は、トライ・インデックスの例を示す図である。この図には、辺節の情報が格納されたアドレス、その辺節の直下の弟ノードの位置（アドレス）、その辺節に対応するラベル、ノードの状態及び対応する文書集合へのリンク情報を示している。この例では、ラベルは漢字1文字である。ノードの状態は、「終了」「継続」のいずれか一方、若しくは双方が設定されている。「終了」は、そのノードに対応する単語が存在することを示し、「継続」は、そのノードが子供を有していることを示す。

【0044】このようにトライを構成することによって、ある「辺節A」が指定されたときに、「辺節A」の長男および「辺節A」のすぐとなりの弟の辺節を直ちに特定できる。

【0045】具体的な事例データを想定して、本実施例の動作を説明しよう。簡単のために、図5に示す単語の集合が与えられ、そのうち「解像度」と「解像力」が互いに関連しているとする。

【0046】まず、すべての単語を表すトライを作成し、単語の末尾に相当する最終ノードには、その単語に関連する単語の集合へのリンクを張っておく。これをトライ1とよぶ。図5の単語集合が与えられた場合であれば、図6のトライを構成する。

【0047】次に、単語の集合へのリンク情報から単語を対応付けるテーブルを用意する。これを関連語対応テ

ープルとよぶことにする。図8は、関連語対応テーブルを示す図である。このテーブルは、単語から関連語集合へのリンク情報とノード位置の集合とが対応付けられている。この例では、「解像度」と「解像力」が互いに関連していることから、リンク情報「T3」「T4」にはノードの位置「14, 19」が対応している。

【0048】次に、本実施の形態の検索を実行する動作を、例を交えて説明する。検索キーとして単語が入力され、入力単語に関連する単語集合を求める場合を考えることにする。いま想定している事例データのもとで、

「解像度」が検索キーとして入力された場合を考える。

【0049】まず、ノード位置検索部22によって、検索キーの単語「解像度」に対応するノード位置の集合、すなわち、関連単語集合へのリンク情報を求められる。具体的には次のように行う。トライから単語を検索する通常の方法によって、トライ・インデックスから「T3」というリンク情報を求める。そして、関連語対応テーブルから、「T3」に対応するノード位置集合として、「14, 19」を得る。

【0050】次に、単語検索部23が、ノード位置検索部22によって得られたノード位置集合から、対応する単語集合を求める。以下に、図6のトライに基づいて、1つのノード位置をキーとして対応する単語を検索する手順を説明する。

【0051】図9は、第1の実施の形態における単語検索手順を示すフローチャートである。

【S11】検索キーのノードアドレスは「A」であたえられたとする。まず、単語の文字列を記録するためのラベル記録部を用意しておく。はじめはトライの先頭にある辺節に注目する。注目している辺節を「現在の辺節」と、その辺節に含まれるアークとノードとを、それぞれ「現在のアーク」、「現在のノード」とよぶことにする。

【S12】現在の辺節が「A」と等しいか否かを判断する。等しければステップS13に進み、等しくなければステップS14に進む。

【S13】現在の辺節のアドレスが、「A」と等しいとき、現在のアークのラベルをラベル記録部にプッシュ（格納）し、ラベル記憶部の内容を出力し、正常に処理を終了する。

【S14】現在の辺節が弟を有しているか否かを判断する。弟を有していればステップS15に進み、有していなければステップS17に進む。

【S15】現在の辺節が弟を有している場合には、隣接する弟の辺節のアドレスが「A」以下か否かを判断する。「A」以下であればステップS16に進み、「A」以下でなければステップS17に進む。

【S16】隣接する弟の辺節のアドレスが「A」以下の場合、隣接する弟の辺節に注目し、ステップS12に進む。

【S17】現在の辺節が子を有しているか否かを判断する。子を有していればステップS18に進み、子を有していなければ不正終了として処理を終了する。

【S18】現在の辺節が子を有している場合には、長子の辺節のアドレスが「A」以下であるか否かを判断する。「A」以下であればステップS19に進み、「A」以下でなければ、不正終了として処理を終了する。

【S19】現在のアークのラベルをラベル記録部にプッシュして、長子の辺節に注目し、ステップS12に進む。

【0052】以上の処理によって、ノード位置をキーとして対応する関連単語が出力される。例として、単語へのリンク情報としてアドレス「14」が得られたとき、上記のアルゴリズムにしたがって図7のように記録されたトライから対応する関連単語を求めることにする。

【0053】まず、ステップS11より、「A」に「14」を代入し、アドレス「0」の辺節に注目する。ラベル記録部は空にしておく。以下、辺節はそのアドレスによって識別することにする。すなわち、アドレス「0」の辺節は、辺節「0」とよぶことにする。ノードとアークに関しても辺節と同様に識別することにする。

【0054】ステップS12において、現在の辺節「0」は「A (=14)」に等しくないので、ステップS14に進む。ステップS14において、現在の辺節「0」には弟の辺節が存在するので、ステップS15に進む。ステップS15において、隣接する弟=24、A=14であり、隣接する弟≦Aは成り立たないので、ステップS17に進む。

【0055】ステップS17において、現在の辺節「0」には子の辺節が存在するので、ステップS18に進む。ステップS18において、長子=5、A=14であり、長子≦Aが成り立つので、ステップS19に進む。

【0056】ステップS19において、現在のアーク「0」のラベル「解」をラベル記録部にプッシュし、長子の辺節「5」に注目する。そして、ステップS12に進む。再びステップS12において、現在の辺節「5」は「A (=14)」に等しくないので、ステップS14に進む。

【0057】ステップS14では、現在の辺節「5」に対して弟の辺節が存在するので、ステップS15に進む。ステップS15では、隣接する弟(=10)≦A(=14)が成り立つので、隣接する弟「10」に注目し、ステップS12に進む。

【0058】ステップS12において、現在の辺節「10」は「A (=14)」に等しくないので、ステップS14に進む。ステップS14において、現在の辺節「10」に対して弟の辺節が存在しないので、ステップS17に進む。

【0059】ステップS17において、現在の辺節「1

0」には子の辺節が存在するので、ステップS18に進む。ステップS18において、長子=14、A=14であり、長子≤Aが成り立つので、ステップS19に進む。

【0060】ステップS19において、現在のアーク「10」のラベル「像」をラベル記録部にプッシュし、長子の辺節「14」に注目する。そして、ステップS12に進む。

【0061】ステップS12において、現在の辺節「14」は「A(=14)」と等しいのでステップS13に進む。ステップS13において、現在のアーク「14」のラベル「度」をラベル記録部にプッシュし、ラベル記録部の内容である「解、像、度」を出力し、正常に処理を終える。

【0062】以上の処理によって、ノードの位置「14」から、対応する単語である「解像度」を得ることができる。同様に、ノードの位置「19」からは、「解像力」が得られる。すなわち、「解像度」の入力に対して、「解像度、解像力」の出力が得られたことになる。

【0063】このようにして、トライを用いて、関連語の検索処理を少ない記憶容量で高速に行うことが可能となる。次に、第2の実施の形態について説明する。第2の実施の形態は、文字からその文字を含むキーワードを検索する単語検索装置である。

【0064】図10は、第2の実施の形態の概略構成を示す図である。この実施の形態に係る単語検索装置は、単語格納部51、文字インデックス部52、ノード位置検索部53、及び単語検索部54とから構成されている。この構成要素のうち、単語格納部51、ノード位置検索部53及び単語検索部54は、図3に示した、単語格納手段11、ノード位置検索手段14及び単語検索手段12の機能を有している。また、文字インデックス部52は、図3のキーインデックス格納手段13のキーインデックスを具体的な文字インデックスとしたものである。

【0065】はじめに、索引単語の集合をトライで表現し、単語格納部51を構成する。索引単語を表す最終ノードには、その索引単語に関連する文書の集合をたどるようにリンクを張っておく。第1の実施の形態と同様に、図5のように与えられた索引単語に対応して図6のトライによる単語インデックス(トライ・インデックス)を構成する。この単語インデックスは、単語格納部51に格納される。

【0066】さらに、索引単語を構成する文字からその索引単語を導くことができる文字インデックスを構成する。この文字インデックスは、文字インデックス部52に格納される。図5の索引単語集合の場合、索引単語を構成する全ての文字の集合は、{解、現、析、像、度、力}である。この文字集合の各文字について、図6のト

ライを先頭からたどることによって、各々の文字の位置を求める。その結果から、文字インデックスを作成する。

【0067】図11は、文字インデックスの例を示す図である。図のように、文字とその文字を含む索引単語を対応させる。この例では、トライ55を文字インデックスのデータ構造として採用している。

【0068】図5から、文字「解」を含む索引単語は「解」「解析」「解像度」「解像力」と複数あることが分かるが、これらへのリンク情報は「解」という文字を表す1つのノードの位置「0」だけで済んでいる。このように、トライ・インデックス中での文字のノードの位置を用いると、単語の末尾のノード位置を単語へのポインタとする場合に比べて、文字インデックスの容量を小さくすることができる。

【0069】さて、検索を実行する動作を例を交えて説明する。検索キーとして1文字が入力され、この文字を含む索引単語と文書集合へのリンクとを求める場合を考えることにする。具体例として、図11および図6のインデックスを用いて、検索キーとして「像」が入力された場合を想定する。

【0070】まず、ノード位置検索部53によって、文字インデックス部52から検索キーの文字(例では「像」)を含んでいる索引単語へのリンクを見つける。図11より、「10」、「28」、「33」というリンク情報が得られることが分かる。

【0071】次に、単語検索部54によって、リンク情報から、単語インデックスを用いて索引単語およびその索引単語が指し示す文書集合を求める。以下に、その手順を説明する。

【0072】図12～図14は、第2の実施の形態において文書集合を求めるための処理手順を示すフローチャートである。図12ではステップS21～S27の処理を示しており、図13ではステップS31～S36の処理を示しており、図14ではステップS41～S48の処理を示している。以下、各ステップの処理内容を説明する。

[S21] 求める索引単語へのリンク情報は、トライの辺節アドレスが「A」で与えられたとする。索引単語の文字列を記録するためのラベル記録部、ラベル記録部に記録した文字列を一時的に待避させておくためのラベルスタック、及びラベルを待避させたときの辺節を保存しておくための辺節スタックをそれぞれ用意し、内容をクリアしておく。開始ノードに移動し、トライの先頭にある辺節に注目する。注目している辺節を現在の辺節とよぶことにする。

[S22] 現在の辺節が、与えられたリンク情報「A」と等しいか否かを判断する。等しければステップS23に進み、等しくなければステップS31(図13に示す)へ進む。

〔S23〕現在の辺節が、与えられたリンク情報「A」と等しい場合、現在のアークのラベルをラベル記録部にプッシュする。

〔S24〕現在のノードが終了状態か否かを判断する。終了状態であればステップS25に進み、終了状態であればステップS26に進む。

〔S25〕現在のノードが終了状態の場合、ラベル記録部の内容と、現在の終了状態ノードに対応する文書集合へのリンクとをそれぞれ出力する。

〔S26〕現在の辺節が子を持っている否かを判断する。子をもっていればステップS27に進み、子をもっていないければ、処理を正常終了する。

〔S27〕現在の辺節が子をもっている場合、長子の辺節に注目し、ステップS41（図14に示す）に進む。

〔S31〕ステップS22において、現在の辺節が弟をもっていると判断された場合、ステップS32に進み、弟をもっていないければステップS34に進む。

〔S32〕隣接する弟の辺節のアドレスが「A」以下か否かを判断する。「A」以下であればステップS33に進み、「A」以下でなければステップS34に進む。

〔S33〕隣接する弟の辺節のアドレスが「A」以下の場合、隣接する弟の辺節に注目し、ステップS22（図12に示す）に進む。

〔S34〕隣接する弟の辺節のアドレスが「A」以下でない場合、現在の辺節が子をもっているか否かを判断する。子をもっていればステップS35に進み、子をもっていないければ不正終了として処理を終了する。

〔S35〕長子の辺節のアドレスが「A」以下か否かを判断し、「A」以下であればステップS36に進み、

「A」以下でなければ不正終了として処理を終了する。

〔S36〕現在のアークのラベルをラベル記録部にプッシュして、長子の辺節に注目する。そして、ステップS22（図12に示す）に進む。

〔S41〕現在の辺節が弟をもっているか否かを判断する。弟をもっていればステップS42に進み、弟をもっていないければステップS43に進む。

〔S42〕現在の辺節が弟をもっている場合には、ラベル記録部の内容をラベルスタックに、隣接する弟の辺節を辺節スタックにそれぞれプッシュする。

〔S43〕現在のノードが終了状態か否かを判断する。終了状態であればステップS44に進み、終了状態であればステップS45に進む。

〔S44〕現在のノードが終了状態の場合、ラベル記録部の内容と現在のアークのラベルをつなげたもの、現在の終了状態ノードに対応する文書集合へのリンクとをそれぞれ出力する。

〔S45〕現在の辺節が子をもっているか否かを判断する。子をもっていればステップS46に進み、子をもっていないければステップS47に進む。

〔S46〕現在の辺節が子をもっている場合、現在のア

ークのラベルをラベル記録部にプッシュして、長子の辺節に注目する。そして、ステップS41に進む。

〔S47〕ラベルスタックと辺節スタックが空か否かを判断する。空でなければステップS48に進み、空であれば正常終了する。

〔S48〕ラベルスタックと辺節スタックが空ではないとき、ラベルスタックからラベル記録部にポップし、辺節スタックからポップした辺節に注目する。そして、ステップS41に進む。

10 【0073】以上の処理によって、索引単語の文字列と、その索引単語から対応する文書集合へのリンクが出力される。例として、索引単語へのリンク情報として「10」が得られたとき、上記のアルゴリズムにしたがって図7のトライをたどることにする。

20 【0074】まず、ステップS21より、「A」に「10」を代入し、辺節「0」に注目する。ラベル記録部、ラベルスタック、辺節スタックはいずれも空にしておく。ステップS22において、現在のノード「0」は「A (=10)」に等しくないので、ステップS31に進む。

【0075】ステップS31において、現在のノード「0」には弟が存在するので、ステップS32に進む。ステップS32において、隣接する弟=24、A=10であり、隣接する弟 $\leq$ Aは成り立たないので、ステップ34に進む。

【0076】ステップS34において、現在のノード「0」には子が存在するので、ステップS35に進む。ステップS35において、長子=5、A=10であり、長子 $\leq$ Aが成り立つので、現在のアーク「0」のラベル「解」をラベル記録部にプッシュし、長子の辺節「5」に注目する。そして、ステップS22に進む。

【0077】再びステップS22において、現在の辺節「5」は「A (=10)」に等しくないので、ステップS31に移る。ステップS31では、現在の辺節「5」に対して弟の辺節「10」が存在するので、ステップS32に進む。ステップS32では、隣接する弟(=10) $\leq$ A(=10)は成り立つので、隣接する弟「10」に注目し、ステップS22に進む。

40 【0078】ステップS22において、現在の辺節「10」は「A (=10)」と等しいので、現在のアーク「10」のラベル「像」をラベル記録部にプッシュする。現時点のラベル記録部の内容は「解像」である。ステップS24に進む。ステップS24において、現在のノード「10」は終了状態ではないので、ステップS26に進む。ステップS26において、現在の辺節「10」は子をもっているため、長子の辺節「14」に注目し、ステップS41に進む。

50 【0079】ステップS41において、現在のノード「14」は弟をもっているため、ステップS42に進む。ステップS42において、ラベル記録部の内容であ

る「解像」をラベルスタックにプッシュし、隣接する弟の辺節「19」を辺節スタックにプッシュする。そして、ステップ43に進む。

【0080】ステップS43において、現在のノード「14」は終了状態なので、ステップS44に進む。ステップS44において、ラベル記憶部の内容である「解像」とアーク「14」のラベルである「度」をつなげた「解像度」と、ノード「14」に対応する文書集合へのリンク情報である「T3」を出力する。そして、ステップS45に進む。

【0081】ステップS45において、現在のノード「14」は子をもたないので、ステップS47に進む。ステップS47において、ラベルスタックと辺節スタックは空ではないので、ステップS48に進む。ステップS48において、ラベルスタックからポップして「解像」を取り出し、これをラベル記録部に代入し、辺節スタックからポップして辺節「19」を取り出し、この辺節「19」に注目する。現時点では、ラベルスタックと辺節スタックは空である。そして、ステップS41に進む。

【0082】再びステップS41において、現在のノード「19」は弟をもたないので、ステップS43に進む。ステップS43では、現在のノード「19」は終了状態なので、ステップS44に進む。ステップS44において、ラベル記憶部の内容である「解像」とアーク「19」のラベルである「力」をつなげた「解像力」と、ノード「19」に対応する文書集合へのリンク情報である「T4」を出力する。そして、ステップS45に進む。

【0083】ステップS45において、現在のノード「19」は子をもたないので、ステップS47に進む。ステップS47において、ラベルスタックと辺節スタックは空なので、正常に処理を終わる。

【0084】以上の処理によって、「像」という文字を含む索引単語へのリンクである「10」から、2つの索引単語（「解像度」と「解像力」）、および対応する文書集合へのリンク情報（「T3」と「T4」）を求めることができる。

【0085】次に第3の実施の形態について説明する。図15は、第3の実施の形態の概略構成を示すブロック図である。これは、任意の正規表現からその正規表現を満たすキーワードを検索する単語検索装置である。

【0086】この実施の形態に係る単語検索装置は、単語格納部61、文字インデックス部62、ノード位置検索部63、単語検索部64、及び正規表現解析部65で構成されている。このうち、単語格納部61、文字インデックス部62、ノード位置検索部63、及び単語検索部64は、図10に示した単語格納部51、文字インデックス部52、ノード位置検索部53、及び単語検索部54とほぼ同じ機能を有している。

【0087】正規表現解析部65は、正規表現の検索キーが入力されると、検索キーを解析し、その検索キーに適合する単語集合を得る。その際、必要に応じて、ノード位置検索部63へ文字列を入力し、その戻り値として各文字列のノード位置を得る。また、ノード位置を単語検索部64に入力して、単語集合を得る。

【0088】以下に、正規表現解析部65が検索を実行する際の動作を説明する。なお、トライにおいて、2つのノード「N1」、「N2」を指定したときに、ノード「N1」からたどることができ、ノード「N2」からはたどることができないような、「N1」を開始ノードとするトライの部分木を、「N1」と「N2」から規定されるサブトライとよぶことにする。

【0089】ここで、以下の3つの手続き関数「F1」、「F2」、「F3」を定義する。これらの関数の処理は、正規表現解析部65で行われる。まず、関数「F1」について説明する。関数「F1」は、サブトライ「T」と正規表現「R」を引数とし、文字列の集合「S」を値として返す関数である。

【0090】図16、図17は、関数「F1」の処理手順を示すフローチャートである。図16は、ステップS51～S58の処理を示し、図17は、ステップS61～S68の処理を示している。

【S51】正規表現「R」の先頭が、確定している文字列「B」または文字集合「B」で始まるか否かを判断する。確定文字列等で始まる場合にはステップS52に進み、確定文字列で始まらない場合にはステップS61に進む。

【S52】正規表現「R」の先頭が確定している文字列「B」または文字集合「B」から始まる場合、「R」から「B」の部分を除く正規表現を「R1」とし、以下の処理を行う。

【S53】文字列または文字集合「B」を入力としてサブトライ「T」をたどることができるとかを判断する。たどることができた場合には、ステップS55に進み、たどることができない場合にはステップS54に進む。

【S54】エラーを関数「F1」の値「S」として返し、処理を終了する。

【S55】「B」から「T」をたどることができた場合、正規表現「R1」が空か否かを判断する。空であればステップS56に進み、空でなければステップS58へ進む。

【S56】正規表現「R1」が空の場合、サブトライ「T」を入力「B」でたどった先が終了状態のノードか否かを判断する。終了状態であればステップS57に進み、終了状態でなければステップS54に進む。

【S57】サブトライ「T」を入力「B」でたどった先が終了状態のノードの場合、文字列「B」を関数「F1」の値「S」として返し、処理を終了する。

〔S58〕正規表現「R1」が空でない場合、文字列「B」を入力としてサブトライ「T」をたどったとき、更にたどることのできる残りのサブトライを「T1」とする。関数「F1」に、引数としてサブトライ「T1」と正規表現「R1」を渡し、文字列「B」と関数「F1」の評価した値である文字列集合の各々とを連結し、得られた文字列集合を値「S」として返す。そして、処理を終了する。

〔S61〕正規表現「R」の中に、確定している文字「C1, C2, C3, ...」が含まれているか否かを判断する。確定している文字が含まれていればステップS62に進み、含まれていなければステップS68に進む。

〔S62〕確定している文字Ci(ただし、i=1, 2, 3, ...)をトライにおいて表しているノードが出現する数をAiとし、A1, A2, A3, ...の中で最小値をとる文字「Cj」を、ノード位置検索部63の出力から決定する。

〔S63〕文字「Cj」を表しているノード位置の各々「Pi(ただし、i=1, 2, 3, ...)」について、正規表現「R」において文字「Cj」を最後に含む「R」の一部を正規表現「R1」とし、残りの正規表現を「R2」とし、「R1」を受理する有限状態オートマトン「M」をつくり、ステップS64に進む。すべてのノード位置「Pi」について処理を終えたとき、ステップS67に進む。

〔S64〕関数「F2」に、引数として有限状態オートマトン「M」、サブトライ「Ti」、ノード位置「Pi」を渡し、関数「F2」を評価した値である文字列「s」を得る。

〔S65〕関数「F2」の値である文字列「s」が正常出力か否かを判断する。正常出力であればステップS66に進み、正常出力でなければ(文字列「s」がエラーの場合)ステップS63に進む。

〔S66〕ノード位置「Pi」からたどることのできるサブトライを「Ti」とする。関数「F1」に、引数としてサブトライ「Ti」と正規表現「R2」を渡し、文字列「s」と、関数「F1」を評価した値である文字列集合「S1」の各々の要素を連結した文字列の集合を求め、文字列集合「S」としてプッシュする。そして、ステップS63へ進む。

〔S67〕文字列集合「S」を出力し、処理を終える。

〔S68〕「R」を満たす文字数「N」を求め、関数「F3」に、引数としてサブトライ「T」と文字数「N」を渡し、関数「F3」を評価した値である文字列集合を求め、文字列集合「S」として出力し、処理を終える。

〔0091〕次に、関数「F2」について説明する。関数「F2」は、有限状態オートマトン「M」とサブトライ「T」、ノード位置「P」を引数とし、文字列「s」

を値として返す関数である。

〔0092〕図18は、関数F2の処理手順を示すフローチャートである。

〔S71〕サブトライ「T」において、ノード位置「P」を指定すると、そのノードに至るまでの経路は一意に定まる。そこで、サブトライ「T」をノード位置「P」に至るまでたどることによって得られるラベル列「L」を、有限状態オートマトン「M」の入力とする。

〔S72〕ラベル列「L」が有限状態オートマトン「M」に受理されるか否かを判断する。受理される場合はステップS73に進み、受理されない場合はステップS74に進む。

〔S73〕ラベル列「L」が有限状態オートマトン「M」に受理される場合、ラベル列「L」を出力し、処理を終了する。

〔S74〕ラベル列「L」が有限状態オートマトン「M」に受理されない場合、エラーを出力し、処理を終了する。

〔0093〕次に、関数「F3」について説明する。関数「F3」は、サブトライ「T」と文字数「N」を引数とし、文字列の集合「S」を値として返す。図19は、関数F3の処理手順を示すフローチャートである。

〔S81〕開始ノードが終了状態であり、かつ文字数「N」として「0」をとることができる場合、空の文字列を文字列の集合「S」にプッシュする。

〔S82〕サブトライ「T」を文字数「N」分だけすべてたどり、たどった先が終了状態のノードとなるラベル列の集合「L」を求め、「L」を文字列の集合「S」にプッシュする。

〔S83〕「S」が空か否かを判断する。「S」が空であればステップS85に進み、「S」が空でないならステップS84に進む。

〔S84〕「S」が空でない場合、「S」を出力し、処理を終了する。

〔S85〕「S」が空の場合、エラーを出力し、処理を終了する。

〔0094〕以上のように関数を定義すると、関数「F1」に、引数として単語インデックスのトライと任意の正規表現を渡し、関数「F1」を評価することによって、与えた正規表現に適合する単語の集合を得ることができる。

〔0095〕具体例として、正規表現「?アイデ?ア?」に合致する単語を探す場合を考えることにする。正規表現において、「?」は0個以上の任意の文字に該当するワイルドカードを意味する。検索の意図は、「アイデア」に対して「アイディア」などといった表記の揺れを考慮し、かつ「アイデ?ア」という文字列を含む単語を検索することである。このように、正規表現に合致する単語を検索できることによって、検索洩れを少なくすることが期待できる。

10

20

30

40

50

【0096】例えば、以下のような単語インデックスが単語格納部61に格納されている場合を考える。図20は、第3の実施の形態におけるトライ66の例を示す図である。この例では、各辺節のラベルとして、カタカナ1文字が与えられている。

【0097】図21は、第3の実施の形態における文字インデックスの例を示す図である。この文字インデックスでは、各文字に対して、「出現数」と「対応するノード位置」とが対応付けられている。

【0098】ここで、関数「F1」に、引数として単語インデックスのトライ「T」と正規表現「?アイデ?ア?」(R)を渡した場合の、関数「F1」の値の評価手順を説明する。

【0099】ステップS51において、正規表現「R」は確定文字列から始まっていないので、ステップS61へ進む。ステップS61において、正規表現「R」の中に、確定している文字(「ア」「イ」「デ」)が含まれているので、ステップS62へ進む。

【0100】ステップS62において、「R」の確定している3つの文字{C1, C2, C3}={ア, イ, デ}を表すノードの出現数をそれぞれ求める。図21の文字インデックスからそれぞれ、「5」「2」「2」であると分かる。

【0101】ステップS63において、これらの中で最小値をとる文字として「イ」をとりあげる。そして、文字「イ」を表しているすべてのノード位置{P1, P2}={10, 120}について、以下の処理を行う。

【0102】ステップS63で、正規表現「?アイデ?ア?」から、「R1」を「?アイ」、「R2」を「デ?ア?」とし、「R1」を受理する有限オートマトンを「M」を作る。

【0103】図22は、「イ」をとりあげた場合の有限オートマトン67の遷移図である。図に示すように、文字列の途中(最初でもよい)で「ア」、「イ」が連続して出現した場合に、終了状態のノードへ遷移する。

【0104】ステップS64で、関数「F2」に、引数として有限状態オートマトン「M」、サブトライ「T」、ノード位置「P1」を渡し、関数「F2」を評価する。さて、関数「F2」の処理(図18に示す)のステップS71において、サブトライ「T1」をノード位置「P1(=10)」に至るまでに得られるラベル列「L」は「アイ」である。

【0105】ステップS72の判断において、「L」は有限状態オートマトン「M」に受理されることが分かる。ステップS73で、ラベル列「L」(=「アイ」)を出力し、関数「F2」の処理を完了する。

【0106】関数「F2」の値は、文字列「アイ」である。したがって、ステップS64では、これを「s」とする。ステップS65で、文字列「s」(=「アイ」)はエラーではないので、ステップS66へ進む。

【0107】ステップS66で、関数「F1」に、引数としてサブトライ「T1」と正規表現「R2」(=「デ?ア?」)を渡し、関数「F1」を評価する。以後、この関数「F1」に関しては関数「F1'」などと表記する。

【0108】図23は、ノード位置P1(=10)からたどることのできるサブトライ「T1」を示す図である。このサブトライ68は、図20のトライ66におけるノード位置P1(=10)の子孫に該当する全ての経路を抽出したものである。

【0109】さて、関数「F1'」のステップS51において、正規表現「R'」の先頭は確定している文字「B'」(=「デ」)から始まっている。ステップS52において、「R1'」を「?ア?」とし、ステップS53へ進む。

【0110】関数「F1'」のステップS53で、文字「B'」(=「デ」)を入力としてサブトライ「T'」をたどることができるので、ステップS55へ進む。関数「F1'」のステップS55の判断において、正規表現「R1'」「?ア?」は空ではないので、ステップS58へ進む。

【0111】関数「F1'」のステップS58で、文字「B'」(=「デ」)を入力としてサブトライ「T'」をたどり、更にたどることのできる残りのサブトライを「T1'」とする。関数「F1」に、引数としてサブトライ「T1'」と正規表現「R1'」(=「?ア?」)を渡し、関数「F1」を評価する。以後、このF1に関してはF1'などと表記する。

【0112】さて、関数「F1''」のステップS51において、正規表現「R''」(=「?ア?」)の先頭は確定文字列から始まっていないので、ステップS61へ進む。

【0113】関数「F1''」のステップS51において、正規表現「R''」(=「?ア?」)の中に確定文字「C1''」(=「ア」)が含まれているので、ステップS62へ進む。

【0114】関数「F1''」のステップS62で、「Cj''」として「C1''」(=「ア」)が相当し、ノード位置の集合{P1'', P2''}={30, 80}が得られる。

【0115】「C1''」(=「ア」)について、関数「F1''」のステップS63以下の処理を行う。関数「F1''」のステップS63で、「R1''」として「?ア」、「R2''」として「?」をとる。「R1''」を受理する有限状態オートマトン「M''」をつくる。

【0116】関数「F1''」のステップS64で、関数「F2」に、引数として有限状態オートマトン「M''」、サブトライ「T''」、ノード位置「P1''」(=30)を渡し、関数「F2」を評価する。

以後、この関数「F2」に関して関数「F2'」などと表記する。

【0117】さて、関数「F2'」のステップS71において、サブトライ「T'」をノード位置「P1'」（＝30）に至るまでに得られるラベル列「L'」を「ア」として、有限状態オートマトン「M'」に入力する。

【0118】ステップS72において、「L'」（＝「ア」）は有限状態オートマトン「M'」に受理されることが分かる。関数「F2'」のステップS73で、ラベル列「L'」（＝「ア」）を出力し、関数「F2'」の処理を完了する。

【0119】関数「F2'」の値「ア」は、文字列である。関数「F1'」のステップS64では、これを「S'」とする。関数「F1'」のステップS65で、文字列「S'」（＝「ア」）はエラーではないので、ステップS66へ進む。

【0120】関数「F1'」のステップS66で、ノード位置「P1'」（＝30）からたどることのサブトライを「T1'」とし、関数「F1」に、引数としてサブトライ「T1'」と正規表現「R2'」（＝「?」）を渡し、関数「F1」を評価する。以後、この関数「F1」に関しては関数「F1'''」などと表記する。

【0121】さて、関数「F1'''」のステップS51において、正規表現「R'''」（＝「?」）の先頭は確定文字列から始まっていないので、ステップS61へ進む。

【0122】関数「F1'''」のステップS61で、正規表現「R'''」（＝「?」）の中に確定文字が含まれていないので、ステップS68へ進む。関数「F1'''」のステップS68で、「R'''」（＝「?」）を満たす文字数「N'''」は無制限である。関数「F3」に、引数としてサブトライ「T'''」と文字数「N'''」を渡し、関数「F3」を評価した値である文字列集合は{「」,「リズム」}であり、関数「F1'''」の値として返す。

【0123】文字列「s'」（＝「ア」）と、関数「F1'''」を評価した値である文字列集合{「」,「リズム」}の各々の要素を連結した文字列の集合は{「ア」,「アリズム」}である。関数「F1'''」のステップS66で、これを文字列集合「S'」にプッシュする。現時点での「S'」の内容は、{「ア」,「アリズム」}である。そして、ステップS63へ進む。

【0124】関数「F1'''」のステップS63以後の処理で、ノード位置「P1」（＝30）についての処理を終えたことになる。次に同様に「P2'」についての処理を行い、再びステップS63に戻ってきて、結果として、「S'」＝{「ア」,「アリズム」,「イ

ア」}を出力する。

【0125】関数「F1'」のステップS58に戻り、文字「B'」（＝「デ」）と関数「F1'''」の値{「ア」,「アリズム」,「イア」}の各々の要素とを連結し、関数「F1'」の値として{「デア」,「デアリズム」,「ディア」}を返す。

【0126】関数「F1」のステップS66に戻り、文字列「s」（＝「アイ」）と関数「F1'」の値{「デア」,「デアリズム」,「ディア」}の各々の要素とを連結し、文字列集合「S」に{「アイデア」,「アイデアリズム」,「アイディア」}をプッシュする。現時点での「S」の内容は、{「アイデア」,「アイデアリズム」,「アイディア」}である。ステップS63へ進む。

【0127】関数「F1」のステップS63において、文字「イ」を表しているノード位置「P2」（＝120）について以下の処理を行う。関数「F1」のステップS64で、関数「F2」に、引数として有限状態オートマトン「M」、サブトライ「T」、ノード位置「P2」を渡し、関数「F2」を評価した値である文字列「s」（＝「ネオアイ」）を得る。

【0128】関数「F1」のステップS65で、文字列「s」（＝「ネオアイ」）はエラーはないので、ステップS66に進む。F1のステップS66で、「P2」（＝120）からたどることのできるサブトライを「T2」とし、関数「F1」に、引数としてサブトライ「T2」と正規表現「R2」（＝「デ?ア?」）を渡し、関数「F1」を評価する。関数「F1」から文字列集合{「デア」,「デアリズム」}が得られ、「S」に{「ネオアイデア」,「ネオアイデアリズム」}をプッシュし、ステップS63へ進む。現時点での「S」の内容は、{「アイデア」,「アイデアリズム」,「アイディア」,「ネオアイデア」,「ネオアイデアリズム」}である。

【0129】関数「F1」のステップS63で、すべてのノード位置「P1」,「P2」について処理を終えたので、ステップS67で、文字列集合「S」（＝{「アイデア」,「アイデアリズム」,「アイディア」,「ネオアイデア」,「ネオアイデアリズム」}）を関数「F1」の値として返し、処理を終える。

【0130】以上のように、単語インデックスのトライと正規表現「?アイデ?ア?」から、単語集合{「アイデア」,「アイデアリズム」,「アイディア」,「ネオアイデア」,「ネオアイデアリズム」}が求まる。

【0131】なお、上記の原理構成若しくは実施の形態は、以下のような変形例が考えられる。図24は、第4の実施の形態の概略構成を示すブロック図である。これは、第1の実施の形態（図4に示す）における関連単語インデックス部21を複数設けたものである。

【0132】この実施の形態では、2つの関連単語イン

デックス部 71, 72 のそれぞれに、関連単語インデックスが格納されている。ノード位置検索部 73 は、単語が入力されると、双方の関連単語インデックス部 71, 72 からノードの位置集合を取得する。そのノードの位置の集合は、どちらの関連単語インデックス部 71, 72 から取得したのかを示す情報と共に、単語検索部 74 に渡される。

【0133】単語検索部 74 は、ノード位置検索部 73 から受け取ったノード位置の集合に基づいて、関連単語インデックス部 71, 72 から関連単語の集合を取得し、出力する。

【0134】図 25 は、第 5 の実施の形態の概略構成を示すブロック図である。これは、第 4 の実施の形態（図 24 に示す）を具体化したものである。この実施の形態では、よみインデックス部 71a と表記インデックス部 72a が設けられている。よみインデックス部 71a は、表記によって表された表記単語の集合を、深さ優先順にノードが記録されるトライ形式で格納していると共に、表記単語と各表記単語に対応する文字列を構成しているよみ単語のノードの位置の集合とを対応付けて格納している。表記インデックス部 72a は、よみによって表された単語の集合を、深さ優先順にノードが記録されるトライ形式で格納していると共に、よみ単語と各よみ単語に対応する表記単語の文字列を構成しているノードの位置の集合とを対応付けて格納している。

【0135】ノード位置検索部 73a と単語検索部 74a とは、第 4 の実施の形態のノード位置検索部 73 と単語検索部 74 と同様の機能を有している。図 26 は、第 6 の実施の形態の概略構成を示すブロック図である。これは、第 4 の実施の形態の関連単語インデックス部 71, 72 をさらに増やしたものである。

【0136】この実施の形態では、多数の関連単語インデックス部 81a, 81b, 81c, ... が設けられている。ノード位置検索部 82 は、単語の入力を受け取ると、各関連単語インデックス部 81a, 81b, 81c, ... から、該当するノードの位置の集合を受け取る。単語検索部 83 は、ノード位置検索部 82 から受け取ったノード位置の集合に基づいて、関連単語インデックス部 81a, 81b, 81c, ... から関連単語の集合を取得し、出力する。

【0137】以上のように、本発明においては、単語の集合を、深さ優先順にノードが記録されるトライ形式にしたがって格納し、単語格納部を構成すると共に、トライにおいて単語を構成しているノードの位置をトライ中の単語を一意に識別できる値として用いることによって、親のノードへのリンク情報を用いることなく、任意のノードを含む経路を特定することができる。また、トライにおける単語を、トライにおけるノードの位置をポインタとして参照することによって、トライ・インデックス以外に別個に単語集合を表すデータは不要になる。

その結果、必要な記憶容量は、従来技術の場合に比べて、著しく少なくて済む。

【0138】なお、上記の処理機能は、コンピュータによって実現することができる。その場合、システム構築支援装置が有すべき機能の処理内容は、コンピュータで読み取り可能な記録媒体に記録されたプログラムに記述されており、このプログラムをコンピュータで実行することにより、上記処理がコンピュータで実現される。コンピュータで読み取り可能な記録媒体としては、磁気記録装置や半導体メモリ等がある。市場を流通させる場合には、CD-ROM (Compact Disc Read Only Memory) やフロッピーディスク等の可搬型記録媒体にプログラムを格納して流通させたり、ネットワークを介して接続されたコンピュータの記憶装置に格納しておき、ネットワークを通じて他のコンピュータに転送することもできる。コンピュータで実行する際には、コンピュータ内のハードディスク装置等にプログラムを格納しておき、メインメモリにロードして実行する。

【0139】

【実施例】本発明の実施例として、必要な記憶容量を、第 5 の実施の形態による場合と、従来技術の場合とを定量的に比較することにする。

【0140】図 27 は、表記の単語とそれに対応するよみの単語の集合との対応関係を示す図である。図中左側に「表記で表される単語」が示されており、右側に「対応するよみ」が示されている。例えば、表記が「A」の場合、「あるふあ」とよむ場合もあれば、「えー」とよむ場合もある。

【0141】図 28 は、よみの単語とそれに対応する表記の単語の集合との対応関係を示す図である。図中左側に「よみで表される単語」が示されており、右側に「対応する表記」が示されている。例えば、よみが「あ」の場合、その表記は、「あ」「ア」「亜」「阿」「在」「有」など多数ある。

【0142】図 27、図 28 には、先頭の 8 語について示しているが、全体では、表記単語は 93, 452 語、よみ単語は 68, 819 語から成る。図 29 は、第 5 の実施の形態におけるインデックス部の情報量を示す図である。このように、本発明を用いて、表記単語とよみ単語の対応データから、表記単語からよみ単語集合へのインデックスおよびよみ単語から表記単語へのインデックスをそれぞれ作成した結果、トライは 1, 257, 579, 0 バイト、ポインタ・テーブルは 494, 085, 0 バイト、インデックス全体は 1, 751, 664, 0 バイトの記憶容量となった。

【0143】そこで、従来技術の説明における第 1 の従来例、すなわち、単語の集合を固定長または可変長の文字列として格納するレコード構造のデータをインデックスとは別個に用意する方法で、同様の機能を果たすための情報を格納した。

【0144】図30は、第1の従来例における情報量を示す図である。第1の従来例では、テキストとその参照テーブルは2, 042, 570. 5バイト、トライは1, 257, 579. 0バイト、ポインタ・テーブルは494, 085. 0バイト、インデックス全体は3, 794, 234. 5バイトの記憶容量となった。

【0145】また、従来技術の説明における第2の従来例、すなわち、トライ・インデックスを単語集合のデータとみなして、単語の末尾に対応するトライ中のノードの識別番号を単語へのポインタとする方法で、同様の機能

を果すための情報を格納した。

【0146】図31は、第2の従来例における情報量を示す図である。第2の従来例では、トライ・インデックスに、2, 469, 996. 5バイト程度、ポインタ・テーブルは494, 085. 0バイト、そしてインデックス全体として2, 964, 081. 5バイトの記憶容量を必要とすると予想できる。予想において、おおよそのトライデータのサイズから親ノードへのリンクに必要なデータ幅は2. 5バイトと仮定し、トライのノードの数 $N$ とし、親ノードへのリンクに必要なデータ容量 $L$ を、 $L = N \times 2. 5$  (バイト) と計算した。

【0147】以上の結果に基づいて、第5の実施の形態と従来例とを比較した。図32は、第5の実施の形態と従来技術との情報量の比較結果を示す図である。

【0148】この比較結果から、本発明の第5の実施の形態は、第1の従来例に比べてトライを除くインデックスは19. 5%、全体のインデックスは46. 2%、第2の従来例に比べてトライは50. 9%、全体のインデックスは59. 1%の記憶容量しか必要としないことが分かる。したがって、本発明によって、必要な記憶容量の著しい削減効果が得られたと言える。

#### 【0149】

【発明の効果】以上説明したように本発明では、単語の集合を、深さ優先順にノードが記録されるトライ形式にしたがって格納し、トライにおいて単語を構成しているノードの位置をトライ中の単語を一意に識別できる値として用いたため、親のノードへのリンク情報を用いることなく、任意のノードを含む経路を特定することができる。また、トライにおけるノードの位置をポインタとして、トライにおける単語を参照するようにしたため、トライ形式の単語集合とは別個に単語集合を表すデータを用意する必要がない。その結果、必要な記憶容量は、従来技術の場合に比べて、著しく少なくて済む。

【0150】また、本発明の単語検索プログラムを記録した媒体では、記録されている単語検索プログラムをコンピュータで実行することにより、ノードの位置からそのノードに対応する単語を含む単語集合の検索を行う単語検索処理を、コンピュータの少ない記憶容量を使用して実行することが可能となる。

#### 【図面の簡単な説明】

【図1】本発明の原理構成図である。

【図2】ノードの位置からそのノードを含む単語または単語の集合を求めるアルゴリズムを示すフローチャートである。

【図3】単語へのキーを入力とする単語検索装置の原理構成図である。

【図4】本発明の第1の実施の形態を示すブロック図である。

【図5】単語の集合の例を示す図である。

【図6】深さ優先順にノードが記録されたトライの例を示す図である。

【図7】トライ・インデックスの例を示す図である。

【図8】関連語対応テーブルを示す図である。

【図9】第1の実施の形態における単語検索手順を示すフローチャートである。

【図10】第2の実施の形態の概略構成を示す図である。

【図11】文字インデックスの例を示す図である。

【図12】第2の実施の形態において文書集合を求めるための処理手順を示すフローチャート（その1）である。

【図13】第2の実施の形態において文書集合を求めるための処理手順を示すフローチャート（その2）である。

【図14】第2の実施の形態において文書集合を求めるための処理手順を示すフローチャート（その3）である。

【図15】第3の実施の形態の概略構成を示すブロック図である。

【図16】関数F1の処理手順を示すフローチャート（その1）である。

【図17】関数F1の処理手順を示すフローチャート（その2）である。

【図18】関数F2の処理手順を示すフローチャートである。

【図19】関数F3の処理手順を示すフローチャートである。

【図20】第3の実施の形態におけるトライの例を示す図である。

【図21】第3の実施の形態における文字インデックスの例を示す図である。

【図22】「イ」をとりあげた場合の有限オートマトンの遷移図である。

【図23】ノード位置P1 (=10) からたどることのできるサブトライ「T1」を示す図である。

【図24】第4の実施の形態の概略構成を示すブロック図である。

【図25】第5の実施の形態の概略構成を示すブロック図である。

【図26】第6の実施の形態の概略構成を示すブロック

図である。

【図 27】表記の単語とそれに対応するよみの単語の集合との対応関係を示す図である。

【図 28】よみの単語とそれに対応する表記の単語の集合との対応関係を示す図である。

【図 29】第 5 の実施の形態におけるインデックス部の情報量を示す図である。

【図 30】第 1 の従来例における情報量を示す図である。

【図 31】第 2 の従来例における情報量を示す図であ

る。

【図 32】第 5 の実施の形態と従来技術との情報量の比較結果を示す図である。

【符号の説明】

1 単語格納手段

2 単語検索手段

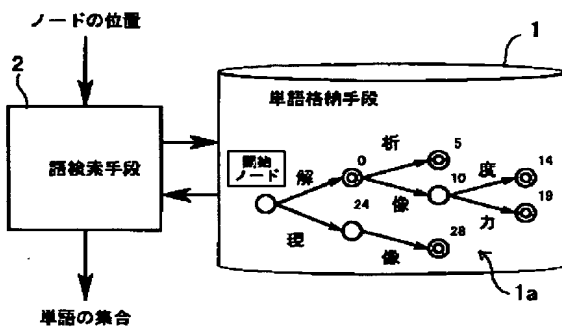
11 単語格納手段

12 キーインデックス格納手段

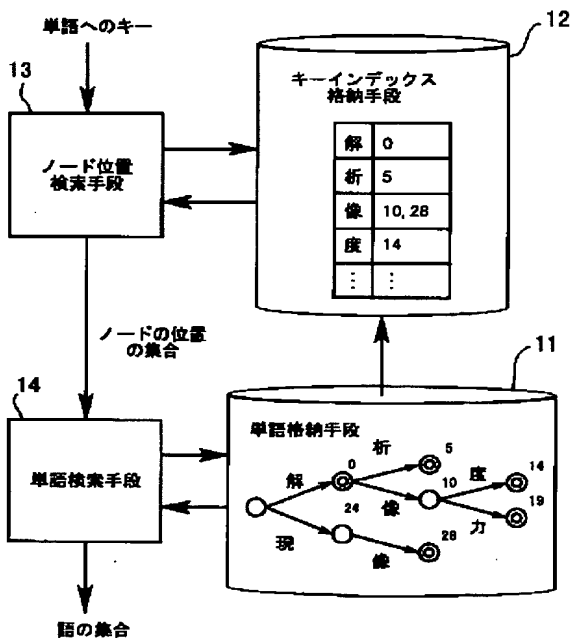
13 ノード位置検索手段

10 14 単語検索手段

【図 1】



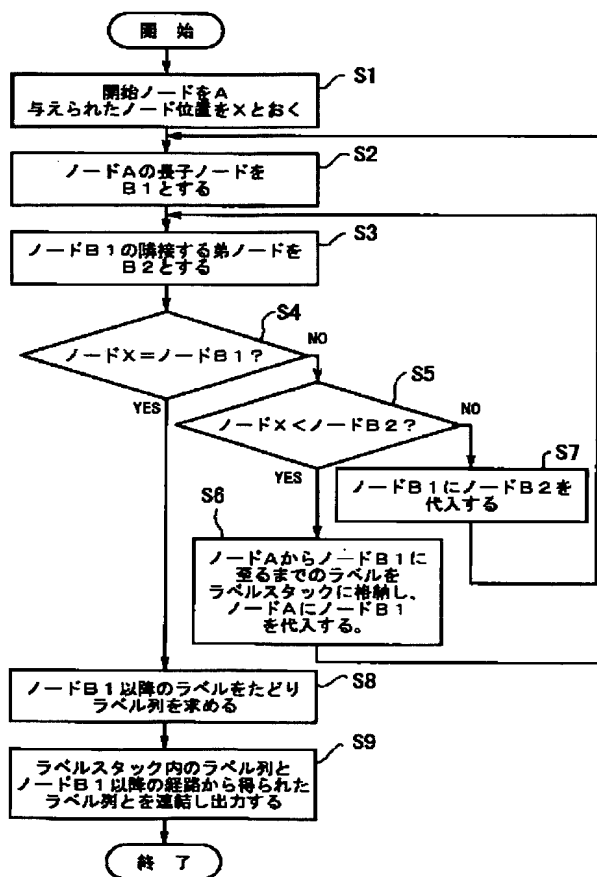
【図 3】



【図 8】

単語からの関連単語集合へのリンク情報	ノード位置の集合
T3	14, 19
T4	14, 19

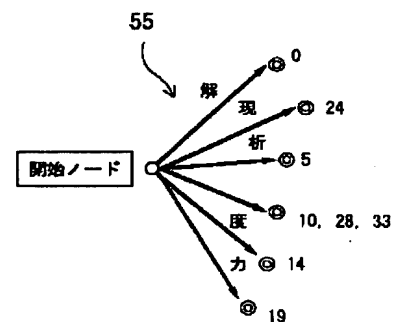
【図 2】



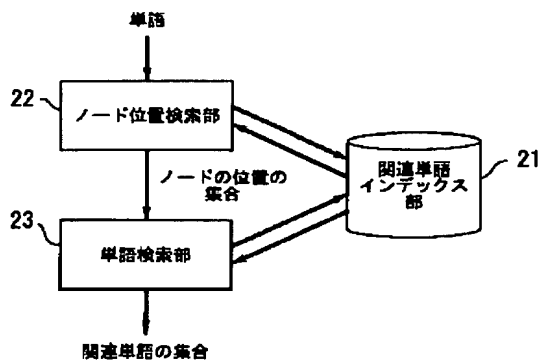
【図 5】

単語	単語に対応するポインタ
解	T1
解析	T2
解像度	T3
解像力	T4
現像	T5
像	T6

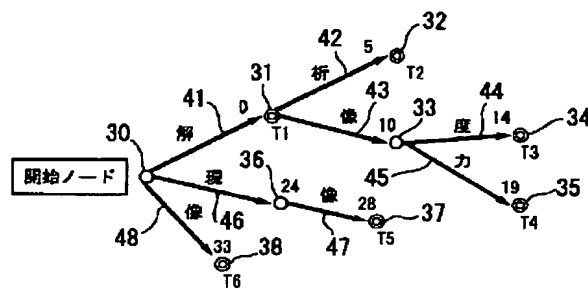
【図 11】



【図 4】



【図 6】

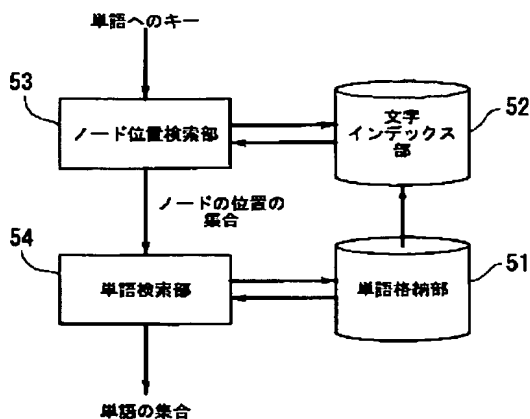


【図 9】

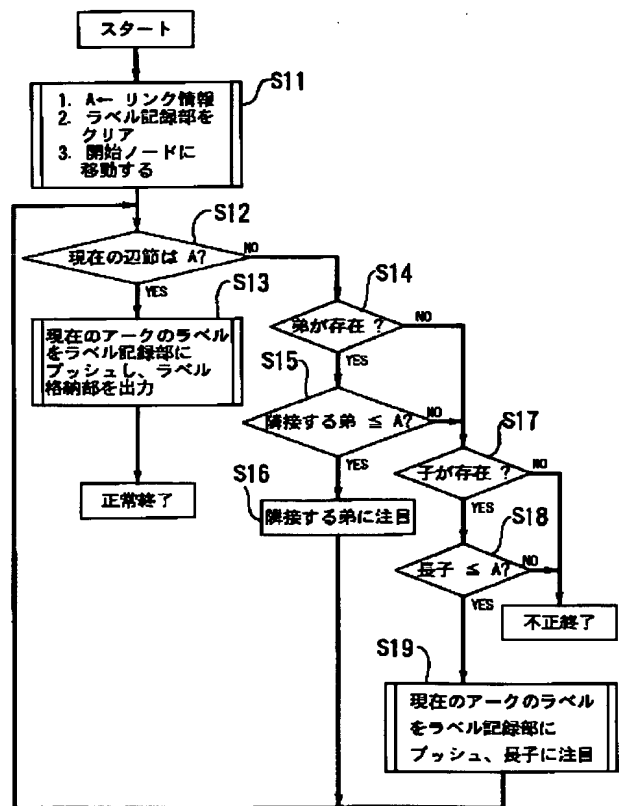
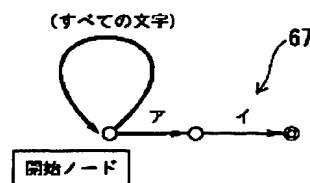
【図 7】

アドレス	直下の弟ノードの位置	ラベル	ノードの状態	リンク情報
0	24	解	(終了・継続)	T1
5	10	折	(終了)	T2
10	-	像	(継続)	-
14	19	度	(終了)	T3
19	-	力	(終了)	T4
24	33	現	(継続)	-
28	-	像	(終了)	T5
33	-	像	(終了)	T6

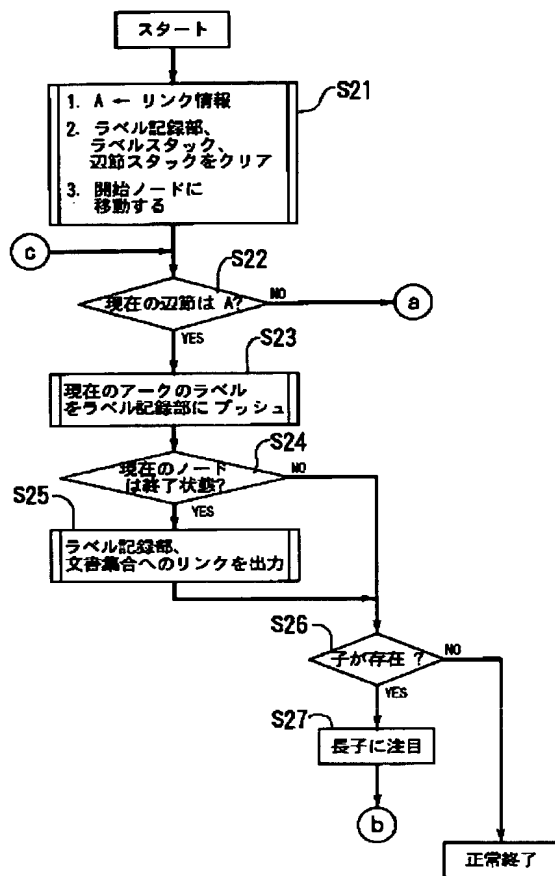
【図 10】



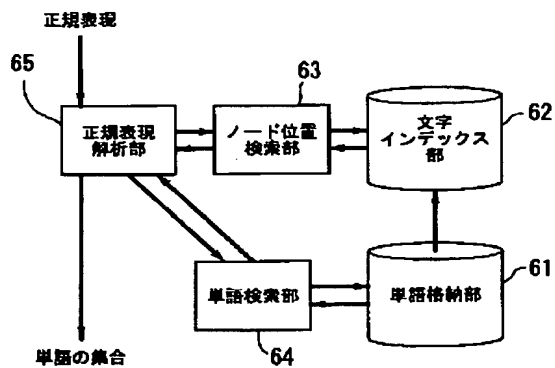
【図 22】



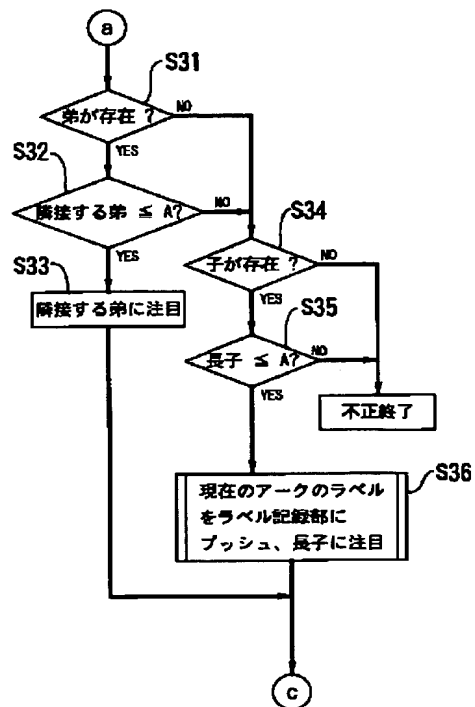
【図 12】



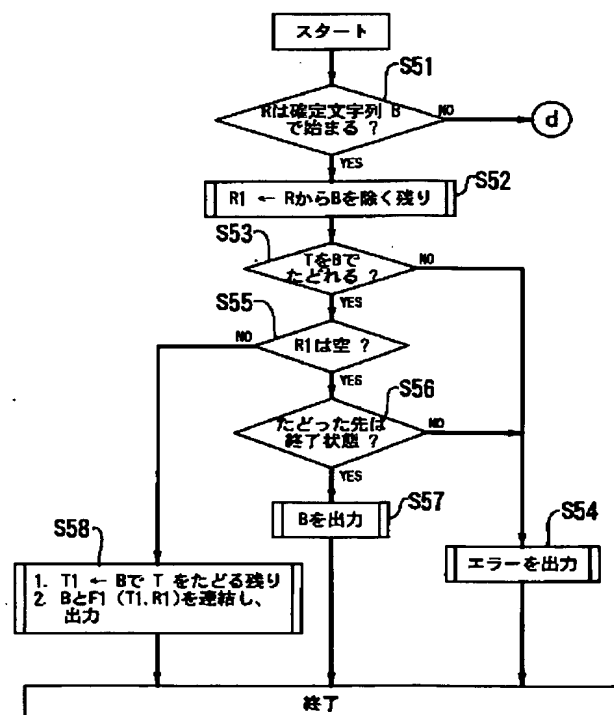
【図 15】



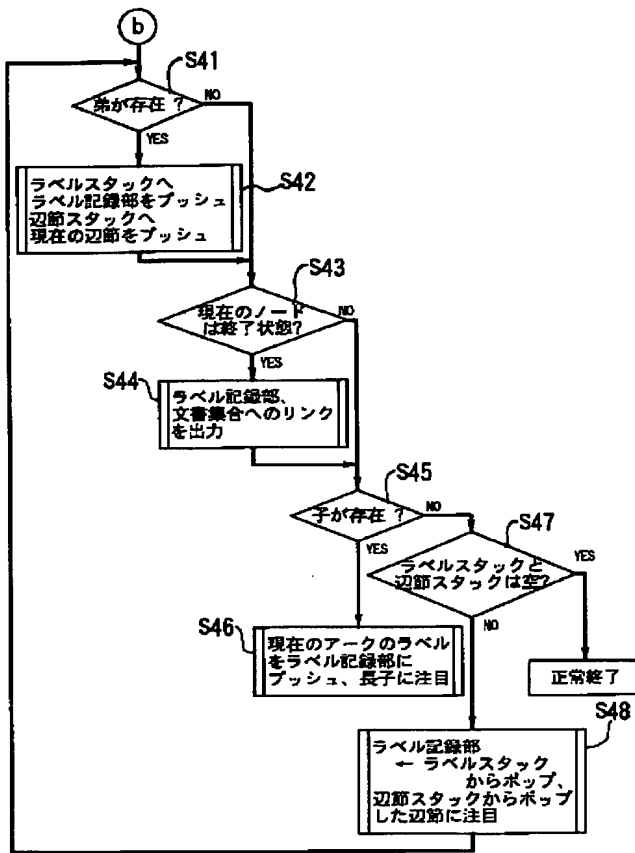
【図 13】



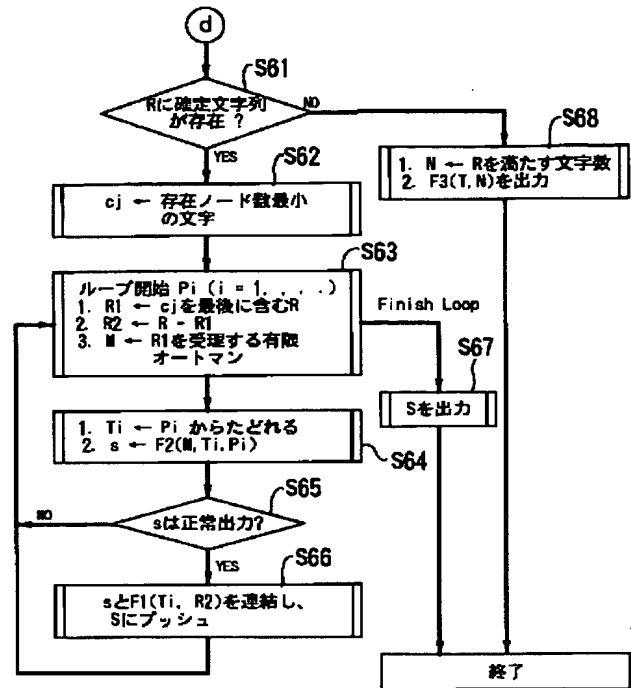
【図 16】



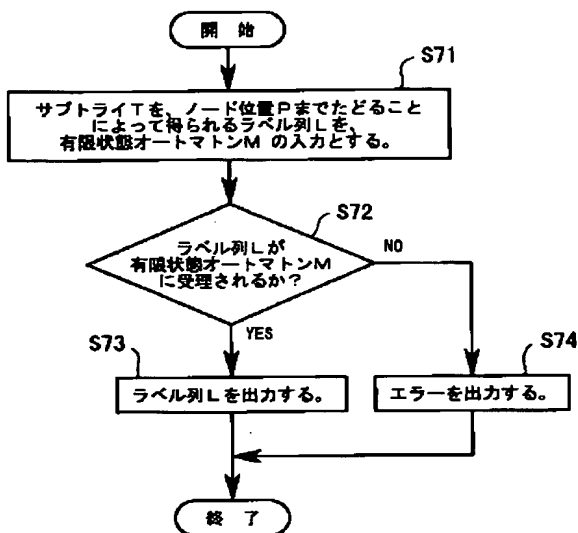
【図 14】



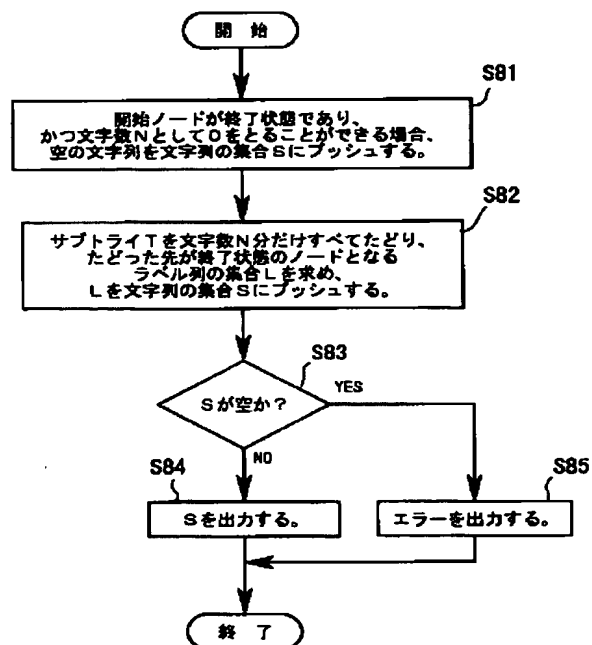
【図 17】



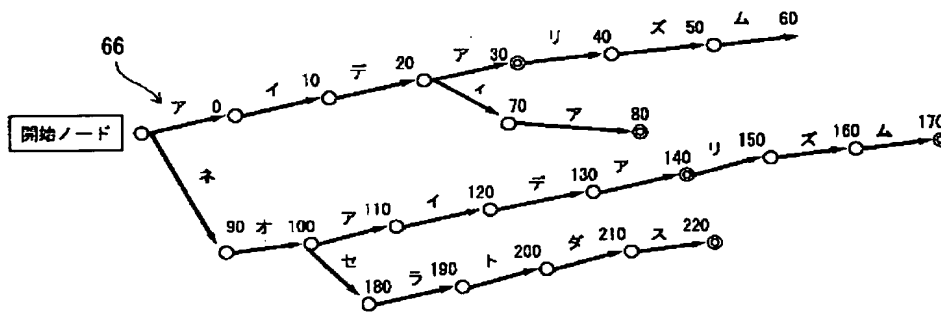
【図 18】



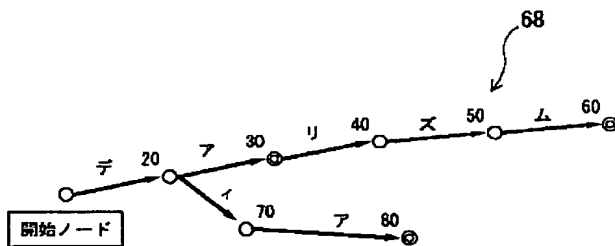
【図 19】



【図 20】



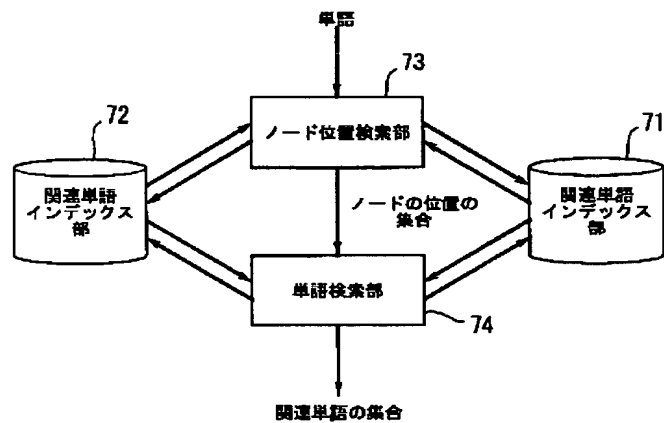
【図 23】



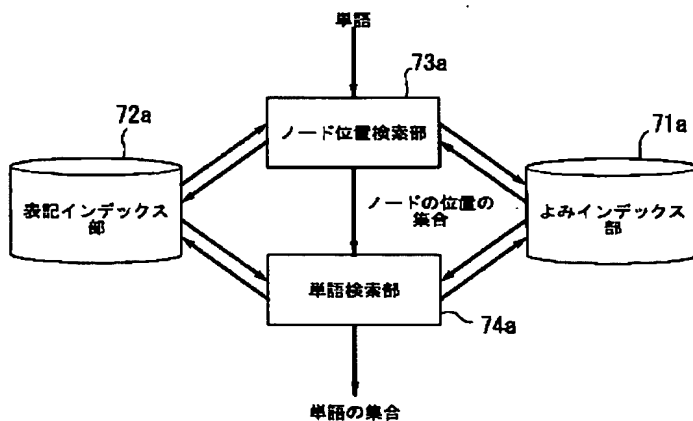
【図 21】

文字	出現数	対応するノード位置
ア	5	0, 30, 80, 110, 140
イ	1	70
イ	2	10, 120
オ	1	100
ス	1	220
ズ	2	50, 160
セ	1	180
ダ	1	210
デ	2	20, 130
ト	1	200
ネ	1	90
ム	2	60, 170
ラ	1	190
リ	2	40, 150

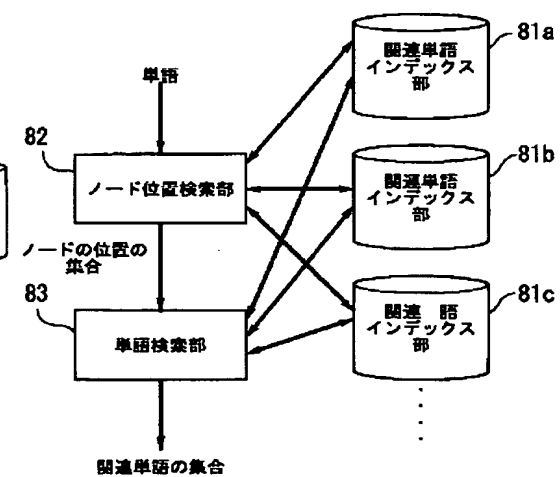
【図 24】



【図 25】



【図 26】



【図27】

表記で表される単語	対応するよみ
A	あるふぁ、えー
ABC	えーびーしー
ABC兵器	えーびーしーへいき
AB型	えーびーがた
AM	えーえむ
AM放送	えーえむほうそう
AP	えーびー
ASA感光度	あさかんこうど、えーえすえーかんこうど
...	...

【図28】

よみで表される単語	対応する表記
あ	あ、ア、亜、阿、在、有
あーく	アーキ
あーくとう	アーキ灯、アーキ燈
あーくらいと	アークライト
あーけーど	アーケード
あーけいっく	アーケイック
あーす	アース
あーち	アーチ
...	...

【図29】

	表記単語	よみ単語
単語数	93,452	68,819
トライのサイズ	688,312.0バイト	569,267.0バイト
トライサイズの合計	1,257,579.0バイト	
ポイントのエントリ数	98817	98817
ポイントの幅	2.5バイト	2.5バイト
ポイント・テーブルサイズ	247042.5バイト	247042.5バイト
ポイント・テーブルサイズの合計	494,085.0バイト	
インデックスの総容量	1,751,664.0バイト	

【図30】

	表記単語	よみ単語
単語数	93,452	68,819
トライのサイズ	688,312.0バイト	569,267.0バイト
トライサイズの合計	1,257,579.0バイト	
テキストのサイズ	978,775.0バイト	658,118.0バイト
テキスト参照位置テーブルの幅	2.5バイト	2.5バイト
テキスト参照位置テーブルのサイズ	233,630.0バイト	172,047.5バイト
テキストと参照テーブルの合計サイズ	2,042,570.5バイト	
ポイントの幅	2.5バイト	2.5バイト
ポイントのエントリ数	98817	98817
ポイント・テーブルサイズ	247042.5バイト	247042.5バイト
ポイント・テーブルサイズの合計	494,085.0バイト	
トライを除くインデックスの合計サイズ	2,536,655.5バイト	
インデックスの総容量	3,794,234.5バイト	

【図 3 1】

	表記単語	よみ単語
単語数	93,452	68,819
トライのノード数	254,184	230,783
予想される親ノードへのリンクサイズ	635460.0バイト	576,957.5バイト
予想されるトライサイズ	1,323,772.0バイト	1,146,224.5バイト
予想されるトライサイズの合計	2,469,996.5バイト	
ポインタの幅	2.5バイト	2.5バイト
ポインタのエントリ数	98817	98817
ポインタ・テーブルサイズ	247042.5バイト	247042.5バイト
ポインタ・テーブルサイズの合計	494,085.0バイト	
予想されるインデックスの総容量	2,964,081.5バイト	

【図 3 2】

	従来技術 A を100 とした 場合の容量比	従来技術 B を100 とした 場合の予想容量比
トライのサイズ	100	50.9
トライを除くインデックスの合計サイズ	19.5	100
インデックスの総容量	46.2	59.1